# CUSTOMER SEGMENTATION USING DATA SCIENCE


## Abstract

The problem is to implement data science techniques to segment customers based on their behavior, preferences, and demographic attributes. The goal is to enable businesses to personalize marketing strategies and enhance customer satisfaction. This project involves data collection, data preprocessing, feature engineering, clustering algorithms, visualization, and interpretation of results.


## Stage 1 : Data collection

Collect customer data:

- Gather customer data from diverse sources such as customer databases , transaction , social media and surveys to enhance segmentation accuracy
- Determine the key variables for effective customer segmentation , considering factors like demographics , purchase history , and online behavior

Data preprocessing and cleaning

- Cleanse and preprocess data to ensure accuracy ,completeness and consistency and to remove errors , duplicates and missing values
- Integrate data from CRM systems , online platforms and other sources to create a comprehensive customer profile

Consent and privacy compliance

- Adhere to data privacy regulations and obtain explicit customer consent before collecting and using their data
- Implement robust security measures , including encryption and access controls to protect customer data

## Stage 2 : Data preprocessing

Label encoding

- Utilize label encoding to convert categorical features (e.g., product categories) into numerical representations.

- Ensure that each unique category is assigned a distinct integer label , facilitating the preparation of data for clustering or classification algorithms.

Transformation of data

- Employ scaling or normalization techniques on numerical features to ensure uniform impact on the analysis.
- Consider method such as standardization (subtracting mean and dividing by standard deviation) or min-max scaling ( scaling values between 0 and 1)

Splitting of data

- Execute a split on the dataset , creating distinct training and testing sets for model evaluation.
- Common split ratios include 70-30 , 80-20 , or 90-10 for training-testing divisions.
- Employ k-fold cross-validation where the dataset is divided into k subsets and the model undergoes training and testing
  k times.

**Stage 3 : Feature engineering**

Feature selection:

- Conduct a comprehensive analysis to pinpoint attributes and features that significantly contribute to the accuracy of customer segmentation . Employ statistical tests , correlation analysis , or domain expertise to discern relevant features

Feature creation

- Aggregate the total spending for each customer based on their transaction history
- Calculate the frequency of customer purchases over a defined period
- Determine the time elapsed since a customer's last purchase.
- Compute the average transaction amount for each customer.
- Augment the segmentation granularity by introducing features that offer a more enhanced understanding of customer behavior.

Encoding categorical variables

- Employ encoding methods like one-hot encoding to represent categorical variables as binary vectors.
- Ensure all categorical features are appropriately encoded to facilitate compatibility with machine learning algorithms
- Validate that all features, including newly created ones, are in a format suitable for machine learning model input.

**Stage 4 : Cluster the data (K-Means clustering algorithm)**

Specify number of clusters(K):

- Determine the optimal number of clusters(K) based on domain-knowledge or data-driven methods like silhouette score or elbow method

Initialize centroids

- Shuffle the dataset and randomly select K data points without replacement to serve as initial centroids.This establishes starting points for the iterative k-means clustering process

Iterative clustering

- Measure distances between data points and centroids, assigning each point to the cluster with the nearest centroid
- Compute new centroids as the mean of all data points assigned to each cluster.
- Iterate assignment and centroid recalculation until convergence, signaled by stable centroids.

**Stage 5 : Visualization**

- Create scatter plots to showcase how clusters are distributed concerning two chosen features. This provides insights into the relationships between clusters and specific attributes.
- Utilize bar charts to represent the sizes or distributions of different customer segments. This provides a clear visual understanding of the composition of each cluster
- Heatmaps to show similarities between segments

**Stage 6 : Interpretation**

Characteristics analysis

- Conduct a detailed analysis of each customer segment to gain insights into their unique characteristics, behaviors, and needs.
- Explore specific features that differentiate one segment from another.
- Consider demographic information, transactional behavior, and any other relevant factors contributing to the distinctiveness of each segment.

Deriving Insights

- Identify key factors that distinguish each customer segment from others. Uncover insights into what makes each segment unique and how they differ in terms of preferences, behaviors, or requirements.
- Utilize the gained insights to customize marketing strategies, refine product offerings, and enhance overall customer experiences.

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from seaborn as sns

from sklearn.preprocessing import StandardScaler, OneHotEncoder

from sklearn.feature_selection import SelectKBest, f_classif

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt


# load the data set

df = pd.read_csv('F:\Mall_Customers.csv')
```

# Split the dataset into features (X) and target variable (y)

X = df[['Genre','Age','Annual Income (k$)']]  # X is provided with the independent variables

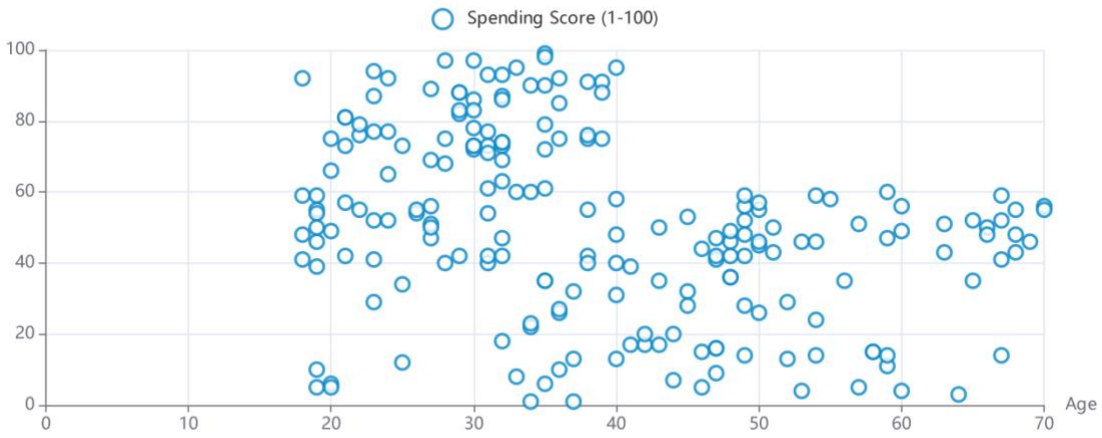
y = df['Spending Score (1-100)'] # y is provided with the dependent variable

# Split the dataset into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42) # here the splitting is done with 80:20 ratio

# Display the relation between the variables

plt.scatter(df['Age'],df['Spending Score (1-100)'])



plt.scatter(df['Annual Income (k$)'],df['Spending Score (1-100)'])


# Create a correlation matrix

correlation_matrix = df.corr()

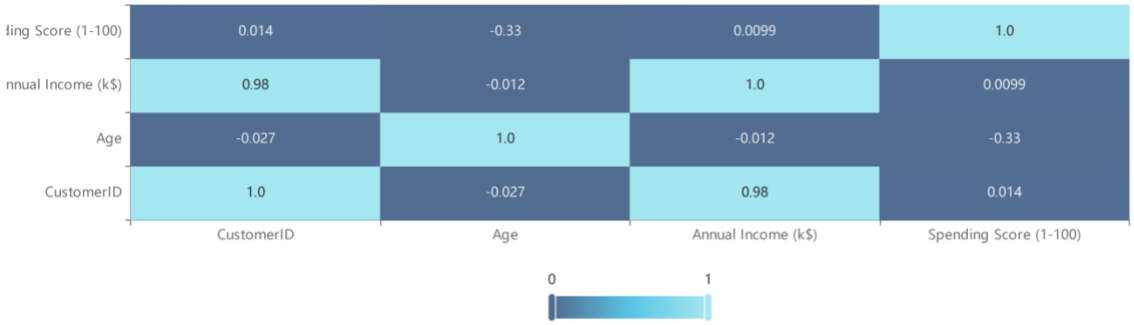
# Create a heatmap

plt.figure(figsize=(12, 10))

sns.heatmap(correlation_matrix, annot=True, cmap='Blues', fmt='.2f', linewidths=0.5)

plt.title('Correlation Heatmap')

plt.show()




# Feature Selection

# Select a subset of features for customer segmentation.

selected_features = df[['Age', 'Annual Income', 'Spending Score']]

# Encoding Categorical Variables

categorical_features = ['Genre']

encoder = OneHotEncoder(drop='first', sparse=False)

encoded_categorical = encoder.fit_transform(df[categorical_features])

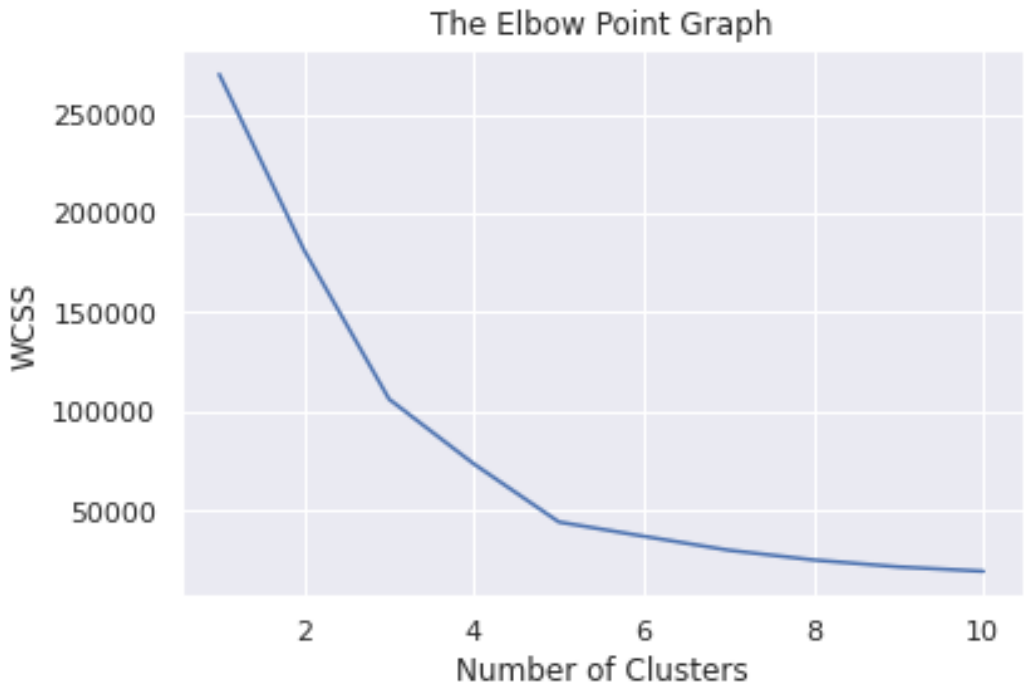
encoded_categorical_df = pd.DataFrame(encoded_categorical, columns=encoder.get_feature_names(categorical_features))

# Concatenate the numerical and encoded categorical features

df_encoded = pd.concat([selected_features, encoded_categorical_df], axis=1)

```
# Feature Scaling

# Standardize the numerical features.

scaler = StandardScaler()

numerical_features = ['Age', 'Annual Income', 'Spending Score']

df_encoded[numerical_features] =
scaler.fit_transform(df_encoded[numerical_features])


# finding wcss value for different number of clusters

#Choosing the Annual Income Column & Spending Score column

X = df.iloc[:,[3,4]].values

wcss = []


for i in range(1,11):

  kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)

  kmeans.fit(X)


  wcss.append(kmeans.inertia_)


  # plot an elbow graph


  sns.set()

  plt.plot(range(1, 11), wcss)

  plt.title('The Elbow Point Graph')
```

```
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```




#training the k means clustering model

```
kmeans = KMeans(n_clusters=5, init='k-means++', random_state=0)

# return a label for each data point based on their cluster
Y = kmeans.fit_predict(X)

print(Y)
```

```
[3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1
3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 0 3 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 2 4 2 0 2 4 2 4 2 0 2 4 2 4 2 4 2 4 2 0 2 4
2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4
2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2]
```

#visualizing all the clusters

```
  plt.figure(figsize=(8, 8))

  plt.scatter(X[Y == 0, 0], X[Y == 0, 1], s=50, c='green', label='Cluster 1')

  plt.scatter(X[Y == 1, 0], X[Y == 1, 1], s=50, c='red', label='Cluster 2')

  plt.scatter(X[Y == 2, 0], X[Y == 2, 1], s=50, c='yellow', label='Cluster 3')

  plt.scatter(X[Y == 3, 0], X[Y == 3, 1], s=50, c='violet', label='Cluster 4')

  plt.scatter(X[Y == 4, 0], X[Y == 4, 1], s=50, c='blue', label='Cluster 5')


  # plot the centroids

  plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=100,
c='cyan', label='Centroids')


  plt.title('Customer Groups')

  plt.xlabel('Annual Income')

  plt.ylabel('Spending Score')

  plt.show()
```


**Observation:**

Purple Group (Low Annual Income - Low Spending Score):

These customers have low annual income and also tend to have a low spending score. They might be more budget-conscious or conservative spenders.

Blue Group (High Annual Income - Low Spending Score):

Customers in this group have high annual income but still exhibit a low spending score. They could be frugal or have specific reasons for not spending much.

## Red Group (Low Annual Income - High Spending Score):

This group consists of customers with low annual income but a high spending score. They might be willing to spend more, despite limited income, which could be an interesting target for marketing or promotions.

## Yellow Group (High Annual Income - High Spending Score):

These customers have both a high annual income and a high spending score, indicating they are high-value customers who are willing to spend more.